

A near real-time framework for extracting tip-sample forces in dynamic atomic force microscopy (dAFM)

David Busch, Qingze Zou, Baskar Ganapathysubramanian*

*Department of Mechanical Engineering, 2100 Black Engineering, Iowa State University,
Ames, IA 50010, USA*

Abstract

The atomic force microscope (AFM) is a versatile, high-resolution tool used to characterize the topography and material properties of a large variety of specimens at nano-scale. The interaction of the micro-cantilever tip with the specimen causes cantilever deflections that are measured by an optical sensing mechanism and subsequently utilized to construct the sample topography. Recent years have seen increased interest in using the AFM to characterize soft specimens like gels and live cells. This remains challenging due to the complex and competing nature of tip-sample interaction forces – large tip-sample interaction force is necessary to achieve good signal-to-noise ratios; However, large force tends to deform and destroy soft samples. In situ estimation of the local tip-sample interaction force is needed to control the AFM cantilever motion and prevent destruction of soft samples while maintaining a good signal-to-noise ratio. This necessitates the ability to rapidly estimate the tip-sample forces from the cantilever deflection during operation.

This paper proposes a first approach to a near real-time framework for tip-sample force inversion. We pose the inverse problem of extracting the tip-sample force as an unconstrained optimization problem. A fast, parallel forward solver is developed by utilizing graphical processing units (GPU). This forward solver shows an effective 30000 fold speed-up over a comparable CPU implementation, resulting in milli-second calculation times. The forward solver is coupled with a GPU based particle-swarm optimization implementation. We illustrate the framework on three classes of tip-sample

*Corresponding author: B. Ganapathysubramanian, Fax: 515 294-3261, Email: baskarg@iastate.edu, URL: <http://www3.me.iastate.edu/bglab/>

interaction inversions. Each of these inversions is performed in sub-second timings, showing potential for integration with on-line AFM imaging and material characterization.

Keywords: Atomic force microscope, Tip-sample interaction, Graphics processing unit, Nano-scale imaging of soft biological sample, Inverse problem, Particle swarm optimization

1. Introduction

The ability to characterize soft materials on the micro/nano-scale has significant implications to several areas in science ranging from fundamental studies in polymer physics [1, 2, 3] to applied bio-engineering [4, 5], where understanding nanoscale behavior and evolution is essential.

Dynamic AFM imaging [6] is a very effective technique to interrogate surface topography of soft samples [1, 7], particularly for live biological samples in their physiologically friendly liquid environment [8, 9].¹ Dynamic atomic force microscopy (dAFM) or intermittent contact mode AFM utilizes a micro-cantilever fixed-free beam to interrogate samples. The cantilever base is driven by a piezo-actuator to oscillate, causing the free tip to tap (i.e., come into intermittent contact with the sample). The oscillation amplitude and phase with respect to the cantilever base are measured and the amplitude is maintained around a set-point value via feedback control. The measured phase and amplitude data are then utilized to construct the sample topography and also related to the material properties of the sample [10, 11].

Although by using dynamic-mode imaging, the detrimental sliding force on the sample has been largely reduced, the applied normal (tapping) force can still be large and result in not only imaging distortion, but more seriously, sample deformation and damage that can completely modify the sample [7]. Large normal force, however, is needed in dynamic-mode AFM imaging to ensure imaging quality (i.e., high signal to noise ratio). The requirement of rapid scanning (high-speed) imaging of specimens further exacerbates these problems [12, 13]. *The challenge in tackling these hurdles lies in the need to maintain a small tip-sample interaction force during the scanning process.*

¹For instance, by using dynamic AFM imaging, time evolving phenomena like crystallization of polymers [2] and the dehydration process of collagen [5, 14] have been experimentally revealed for the first time.

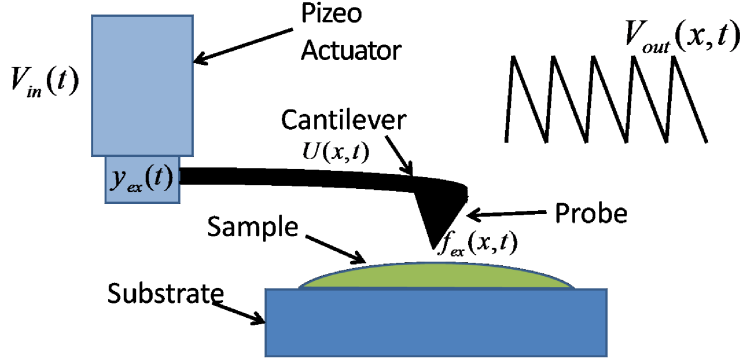


Figure 1: AFM diagram

Therefore, as a key first step to tackle this challenge, estimating the tip-sample interaction force – accurately and in real-time – is essential.

Current methods for tip-sample force inversion generally require significant post-processing time and are, thus, incapable of addressing sample deformation and destruction in real-time [15]. Off-line inverse problems have been formulated to estimate tip-sample interaction forces using conjugate gradient optimization [16] with limited success. The availability of newer computing methods, such as general purpose graphical processing unit computing, opens up the possibility of near real-time inversion. This paper focuses on formulating and implementing a parallel computational framework for fast inversion of tip-sample forces by using the hardware and software capabilities of GPU's and Compute Unified Device Architecture (CUDA), respectively. To the authors best knowledge, this is the first time that near real-time (sub-second) inversion of tip-sample forces has been showcased. Other contributions include: 1) formulating the problem of estimating the tip-sample interaction force as an inverse problem posed as an unconstrained optimization problem; 2) developing an ultra-fast predictive model for AFM dynamics based on parallel algorithms implemented on GPUs; 3) applying gradient-free optimization techniques to quickly find a solution to the optimization problem; and 4) showcasing a hierarchy of models for inversion.

In Section. 2 we formulate the problem definition and pose the direct and inverse problems. Section. 3 details the computational developments and algorithms along with some timing and complexity studies. In Section. 4 we showcase the fast inversion framework on several examples of increasing complexity.

2. Problem formulation

2.1. Introduction to the problem

The physics of the sensing process demonstrates the difficulty in extracting the tip-sample interactions from the measured deflection data. A schematic of the AFM sensing process is shown in Fig. 2.

$$V_{in}(t) \rightarrow y_{ex}(t) \rightarrow f_{ex}(x, t) \rightarrow U(x, t) \leftrightarrow V_{out}(x, t)$$

Figure 2: Dynamic atomic force microscope force interaction process.

An AC input voltage ($V_{in}(t)$), usually a sinusoidal wave is sent to a piezo-electric actuator attached to the base of the cantilever (see Fig. 1), resulting in the oscillation of the base of the cantilever, $y_{ex}(t)$. Then as the vibrating tip is brought into intermittent contact with the sample, the tip-sample interaction force ($f_{ex}(x, t)$) is induced, which in-turn, results in the change of the oscillation (or vibration) pattern at the cantilever tip, $U(x)$. The tip deflection is measured using an optical sensing scheme.

We approach the problem of extracting the tip-sample interaction force $f_{ex}(x, t)$ in two stages— first solve the forward dynamics problem of calculating cantilever deflections given a tip-sample interaction force and subsequently solving associated inverse problem of calculating interaction forces given a cantilever deflection.

2.2. The forward problem: Quantification of tip force effects on AFM cantilever

The forward problem quantifies the relationship between the cantilever base displacement $y_{ex}(t)$ and the tip sample interaction $f_{ex}(x, t)$ with the tip deflection $U(x, t)$. We model the cantilever dynamics by using Euler-Bernoulli (EB) beam theory. The choice is driven by the following rational: (a) the EB model provides a more accurate description of AFM cantilever dynamics than the conventional simple harmonic oscillator model [17]; (b) from an inverse problem standpoint the EB model is computationally more tractable than finite element formulations [15, 17], with minimal loss in fidelity [13, 18]; and (c) all assumptions made in EB beam theory is satisfied by an AFM cantilever [19, 20].

By the EB theory, the cantilever dynamics is modeled as [21]:

$$EIu''''(x, t) + 2\zeta\dot{u}(x, t) + \mu\ddot{u}(x, t) = f_{ex}(x, t) - \mu\ddot{y}_{ex}(t), \quad (1)$$

where u is the relative cantilever displacement with respect to the cantilever base; E , I , μ are the Young's modulus, moment of inertia, and mass per unit length, respectively; ζ is the viscous damping coefficient; f_{ex} is the interaction force, and y_{ex} is the cantilever base displacement. It is a standard practice to convert this PDE into a set of ordinary differential equations (ODE) [21] by solving the free vibration eigenvalue problem and utilizing the orthogonality of the modal/eigen functions, Φ_i . The ODEs are solved for η_i to compute the cantilever deflection $u(x, t) = \sum \Phi_i(x)\eta_i(t)$. This results in a coupled set of ODEs for the coefficients of the modal functions,

$$\ddot{\eta}_i(t) + 2\zeta\omega_i\dot{\eta}_i(t) + \omega_i^2\eta_i(t) = F_i(t) - Y_i(t), \quad (2)$$

where ω_i are the corresponding modal resonance frequencies, $F_i(t) = \langle f_{ex}(x, t), \Phi_i(x) \rangle$ ², and $Y_i(t) = \langle \mu\ddot{y}_{cal}(t), \Phi_i(x) \rangle$.

Solving Eq. (2) and hence the original beam dynamics equation (1) to obtain the cantilever displacement $U(x, t)$ requires that the tip-sample interaction force to be known. Tip-sample interactions are usually parametrized to account for different types of forces. The simplest tip-sample interaction is an elastic response that can be modeled by Hooke's law [22]:

$$f_{spring} = -k(u - h) \quad (3)$$

where $u - h$ is the distance the cantilever tip has pressed into the sample (h is the datum), and k is the local stiffness of the sample. More complex materials respond in a visco-elastic manner, dissipating some of the energy of the tip-sample interaction [15]. This response is modeled using a spring-damper system given by:

$$f_{spring-damper} = -k(u - h) - \zeta_s\dot{u} \quad (4)$$

² $\langle A(x), B(x) \rangle$ is defined as the inner product of A and B over the length of the cantilever, $\langle A(x), B(x) \rangle = \int_0^L A(x)B(x)dx$

where ζ_s is the viscous damping constant. More complex models can include Hertzian contact and Van der Waals forces [15].

In summery, the definition of the forward problem is as follows:

FP: *Given the cantilever properties, (E, I, μ) , the cantilever base displacement $y_{ex}(t)$, and the parametrized tip-sample interaction force, (Eqns. 3 or 4), calculate the cantilever deflection $U(x, t)$.*

2.3. The Inverse problem: Quantification of the tip-sample interaction force

The inverse problem is defined as follows:

IP: *Given the cantilever properties, (E, I, μ) , the measured cantilever tip deflection $U(x, t)$, and the given cantilever base displacement $y_{ex}t$, calculate the parametrized tip-sample interaction $f_{ex}(x, t)$.*

One approach to solving the inverse problem is to convert it into an unconstrained optimization problem through the minimization of a chosen cost functional \mathcal{J} that minimizes the difference between $U(x, t)$ and $u(x, t)$. An appropriate choice of the cost functional \mathcal{J} acts as a metric that quantifies the mismatch between a guess value of the tip-sample interaction and the true tip-sample interaction. The choice of the cost functional plays a significant part in the accuracy and efficiency of the inversion process. The proper choice of the cost functional ensures reasonable speed of calculation and a smooth phase space. Extensive computational experiments suggested the use of the following cost functional:

$$\mathcal{J}_{L2}^2(F) = \int_0^{t_{max}} [U(x, t) - u(x, t)]^2 dt \quad (5)$$

where U is the experimentally measured tip deflection. u is the calculated tip deflection for given the tip-sample interaction F (i.e., by solving the forward problem **FP**).

The unconstrained optimization problem is posed as follows:

Given cantilever properties (E, I, μ) , cantilever base movement $y_{ex}(t)$, and the experimental cantilever tip deflection $(U(x, t))$, find the parametrized tip-sample interaction F^ such that $\mathcal{J}_{L2}(F^*) \leq \mathcal{J}_{L2}(F)$ for any F , where \mathcal{J}_{L2} is defined in Eqn. 5.*

3. Computational framework for near real time inversion

This section details the computational framework for solving the direct and inverse problems formulated in the previous section. A key challenge is

the necessity of very fast force-inversion for real time dAFM imaging of soft samples to be possible. Posing the direct problem as a set of ODE's and the force inversion as an unconstrained optimization problem over these ODEs allows to leverage the computational advantages provided by GPUs. The rational for using GPU's is guided by the following reasons: (1) ability to construct a large set of forward problems in parallel; (2) faster analysis given faster memory accesses compared to CPUs; (3) GPU compute architecture is well suited for problems with minimal parallel dependencies; and (4) GPU compute architecture is well suited for problems where the computation-to-memory-access ratio is larger than one. A brief description of GPU hardware and CUDA software concepts utilized in the developed framework follows.

3.1. CUDA and GPU computing

Utilizing GPU's for computation is different than on CPU's. GPU's require a large number of threads of execution that are processed in parallel to be efficient. In contrast, CPU's are generally more efficient with few threads.

Memory: GPUs (running CUDA) have very large computation capability compared to the speed at which they can access memory. GPU's hide this memory latency by performing computation and memory grabs simultaneously. While sets of threads (called warps) are waiting for their data from memory, other warps get computed. The availability of hierarchies of memory allows significant room for designing algorithms to optimize memory access, thus enhancing speed. We briefly describe the memory modes that are used in the current formulation; *global*, *shared*, *texture*, and *constant*.

1. Global memory is the main memory storage on GPU and is the slowest to access requiring hundreds of clock cycles. Global memory is retrieved in groups of bytes for warps based on the requirements of the threads. Warps can only grab memory that is in sequential order.³
2. Shared memory is a very fast, small block of memory (16 kb on compute capability 1.3 and below, up to 48 kb for compute capability 2.0) which is accessible only within each block of threads.
3. Texture memory is a cached global memory.

³So if thread 0 needs memory from array position 0 and thread 1 requires array position 1000000, the warp will request two accesses to global memory (costing several hundred clock cycles twice). Alternatively, if the memory in array position 1000000 was in position 1 instead, only one memory access would be required.

4. Constant memory can only be assigned by the CPU and is a cached read-only memory.

Given the finite memory resources and speed, memory management is critical as most GPU algorithms are limited by their memory throughput [23].

Computation: Through the use of CUDA architecture and programming tools, the management and control of GPU computation and data parallelism is possible⁴. In CUDA, threads are organized into blocks which are executed on the same streaming multi-processor (SMP). Each GPU only has a finite number of SMPs and as a result can only compute a finite number of blocks at the same time. SMPs execute threads in groups (or warps). Warps are chosen to be processed based on the availability of the requested memory resources. Thus, optimally choosing threads and threads per block can significantly enhance memory access and performance.

Communication: Any data dependency between threads requires special considerations. Shared memory is the best method of dealing with any data dependencies. This means that inter-thread communication is best handled within each block. Communication between blocks can occur through a global sync between all GPU and CPU threads but is very inefficient. Through the use of good parallel programming practices, the shared memory can be used efficiently to communicate between threads.

We utilize CUDA programming to implement both the forward and inverse problems on GPU's. *Our approach takes advantage of the GPU compute structure by designing an algorithm which minimizes data dependency between threads, maximizes the number of computations per global memory access, and minimizes CPU/GPU data communication.* The optimization problem approach allows for minimal data dependency through the solving of many forward problems, that are independently solved on multiple threads on the GPU.

3.2. Solving the forward problem

Fast calculations of solutions to Eqn. 2 are achieved through a CUDA based high-order ODE solver. The three key stages in solving the forward

⁴CUDA provides a compiler and basic functions to perform computational tasks. The CUDA tool-kit also includes a best practices guide which describes the advantages and limitations of GPU computing and how to get the best performance.

problem (FP) are: initialization, including memory set-up and modal function calculation; calculation of modal coefficients via high-order ODE solvers; and using modal functions and modal coefficients to calculate displacements.

Computational issues. A Newton root solver and Simpson integration modules are used to solve for the modal resonance frequency (ω_i) and normalization factor (to make the modal functions orthogonal), respectively. Calculating the modal functions using the hyperbolic trigonometric form generally presented in texts [21] causes over-flow errors for higher modes. We recast the calculation to the following equivalent exponential form to enable accurate calculation without overflow:

$$\Phi_i(x) = \frac{2 \sin(\beta_i x) - 2G \cos(\beta_i x) + (G - 1) \exp^{\beta_i x} + (G + 1) \exp^{-\beta_i x}}{2}, \quad (6)$$

$$G = \frac{2 \exp^{-\beta_i L} \sin(\beta_i L) - \exp^{-2\beta_i L} + 1}{2 \exp^{-\beta_i L} \cos(\beta_i L) - \exp^{-2\beta_i L} + 1}, \quad (7)$$

where β_i is the i^{th} solution of $\cos(\beta L) \cosh(\beta L) + 1 = 0$ and $\omega_i = \beta_i^2 \sqrt{\frac{EI}{\mu L^4}}$.

Note that ω_i^2 grows very quickly with increasing mode order i , making Eqn. 2 a stiff ODE. We utilize an explicit 4th order Runge-Kutta (RK) method to solve the ODEs.⁵ Explicit schemes were tested because they are efficient and the RK fourth order method converged in the range of time steps typical for AFM experiments.

Memory allocation. Solutions of the forward problem are obtained by calculating the modal functions (Φ_i) and the modal coefficients (η_i). The modal functions remain invariant and thus only need to be calculated once. We calculate and store the modal functions serially as part of the initialization process. Two parallel strategies are implemented to assist in solving Eq. 2 for $\eta_i(t)$: *parallel prefix for force inner products and direct parallelism across modes*. Moreover, assuming that the force is variable separable (into spatial and temporal components), the spatial component of the force can be determined during initialization. This allows converting the computation of

⁵First order explicit Euler method was tested but failed to converge to a solution with practical time-step size of greater than 10^{-8} . Second or third order methods have not been tested and could be a possible method of reducing calculation time if they converge.

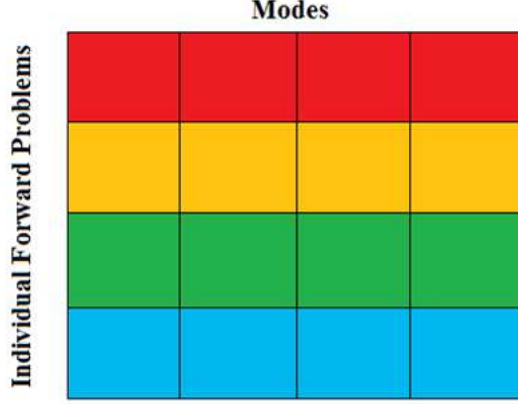


Figure 3: Memory Allocation and Parallelization: A schematic of the forward problem parallel algorithm. Squares represents an individual thread. Each row represents an individual forward problem. Each mode of each forward problem is given a thread. Thus, each block will solve several forward problem solutions simultaneously.

integrals involved on the RHS of Eqn. 2 to a one time calculation. Pre-calculating force integrals leaves implementing a framework with parallelism across modes. Every modal coefficient ODE solve is handled by a unique thread. Thus, two dimensional blocks of threads are set-up as $(m, n_{f_{pb}})$, where m is the number of modes and $n_{f_{pb}}$ is the number of forward problem solutions per block, as shown in Fig. 3. For example, on compute capability 1.3 GPUs, using eight modes, the current parallel framework can run up to 64 problems per block.

We next analyze the memory complexity of the framework. This elucidates the rational for deploying the various data structures in the available memory hierarchies. The major memory needs are as follows:

- The cantilever parameters, E, I, μ . Since the cantilever parameters are assigned as part of the initialization process and require little memory, they are a good choice for constant memory.
- Parameters of the tip-sample interaction. This requires 4 bytes for each parameter, thus requiring $8n_{fps}$ bytes for n_{fps} simultaneous forward problem solutions using the visco-elastic model Eqn. 4. Interaction parameters are left in the global memory since they have to be optimized

in the inverse problem⁶.

- Modal coefficients η_i require $4mn_{fps}$ bytes, where m is the number of modes used. The output deflection points require $n_{dp}n_{fps}$ bytes, where n_{dp} is the number of deflection points computed. Usually deflections at three points on the AFM cantilever are measured ($n_{dp} = 3$). The modal coefficients and deflection points are stored in shared memory because of the constant updating during the forward solve.

Complexity analysis. For the forward problem, the main calculations affecting runtime complexity are the force integral calculations and the ODE solve. The runtime complexity for the serial forward problem is $\mathcal{O}([n_t n_x + n_t]m)$, where m is the number of modes, n_t is the number of time steps and n_x is the number of spatial points used to compute the force integral. However, assuming variable separation of the forces reduces serial runtime complexity to $\mathcal{O}(mn_t)$. By deploying across m threads on a GPU, the parallel runtime complexity is $\mathcal{O}(n_t)$.

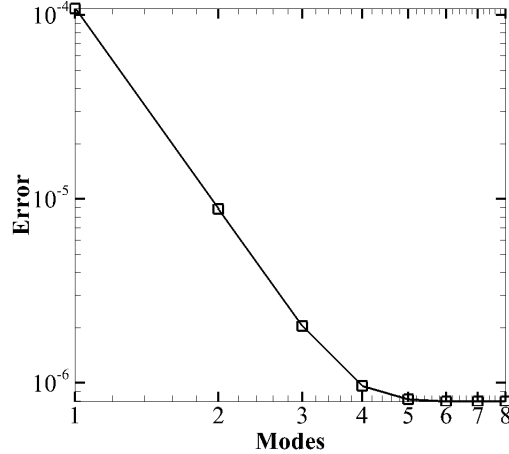


Figure 4: Convergence compared to 8 modes for simple sinusoidal base movement

⁶The access speed of the interaction force parameters could potentially be improved by utilizing texture memory but has not been implemented in this work.

3.3. Speedup characteristics: Comparing CPU vs GPU implementation

In addition to the GPU based implementation discussed in the preceding section, we also implemented a CPU based version of the ODE solver for comparison. The validity of the results of both implementations are ensured by comparing with analytical solutions that are obtained using a sinusoidal base movement and constant forces [21](see Appendix A).

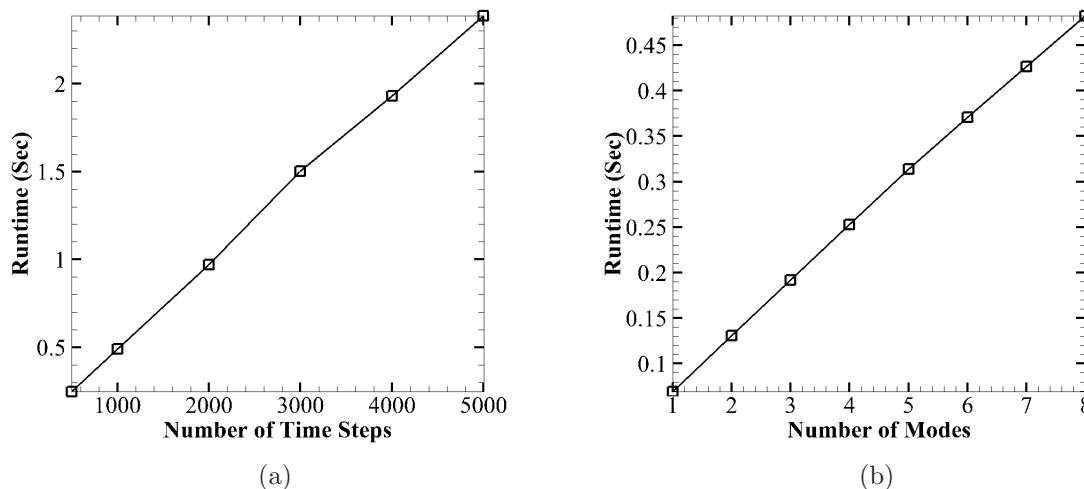


Figure 5: (a) Number of time points vs. runtime in seconds for CPU framework; (b) Number of modes vs. runtime in seconds for CPU based framework

We discuss runtime trends and accuracy details in this subsection. Each forward problem is run for 1000 time steps (unless otherwise stated) using time-step, $\Delta t = 10^{-7}$. A maximum of eight modes are used. Eight modes can satisfactorily track the cantilever evolution (with an error of 3.310^{-6}). Error is defined as $\frac{|V_t - V_c|_{L2}}{|V_t|_{L2}}$, where V_t is the true value and V_c is computed. In testing convergence, the L2 error of various modes are compared to the solution obtained using eight modes. Fig. 4 shows the plot of error versus number of modes. For this simple case, four modes are sufficient for resolving a sinusoidal deflection.

While considering runtime complexity three parameters are most dominant; number of modes, number of points used to describe modal functions, and number of time points. We analyze all three of them independently, first for the CPU based implementation and subsequently for the GPU based implementation. Runtime as a function of the number of modes is shown in

Fig. 5(b) with the number of time points fixed at 1000 (note that 8 modes take 0.482 seconds). Fig. 5(a) plots runtime as a function of the number of time points. When seeking real-time inversions, runtime must be of the order of few hundred milliseconds. These runtime analyses show that calculation times using CPUs are too slow to solve the inverse problem in real time.

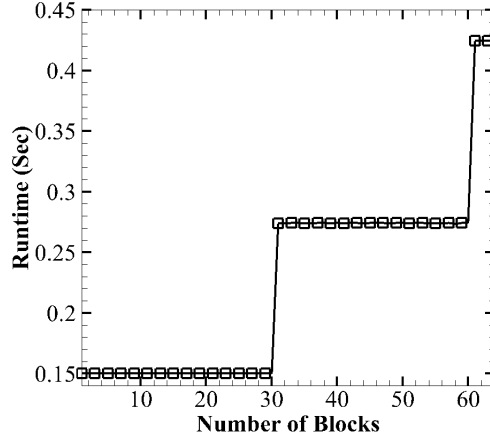


Figure 6: Number of blocks vs. runtime in seconds

In contrast to the CPU results, the GPU based results are very promising. With the goal of solving several hundreds of forward problems to solve one inverse problem, we utilize two metrics to illustrate the capabilities of the GPU based framework: (a) runtime of individual forward solves; (b) the number of forward solves which can be calculated in parallel. Both metrics depend on multiple factors: number of time steps (n_t), number of modes (m), number of solutions per block ($n_{f_{pb}}$), and total number of forward problem solutions ($n_{f_{ps}}$). A Nvidia Quadro FX 5800 GPU is used which limits the number of blocks that can run in parallel to 30 (2 blocks per streaming multiprocessor(SMP), 15 SMP's)⁷. Beyond this, with all SMPs filled, the blocks have to wait for an open SMP. This can be clearly seen in Fig. 6 where the runtime jumps after 30 and 60 blocks. Fixing the number of blocks ($\frac{n_{f_{ps}}}{n_{f_{pb}}}$) to 30 and modes (m) to 8, runtime is analyzed by varying the number of time steps and number of solutions per block. Fig. 7(a) show the

⁷For more details on the system used for testing and implementation, see appendix C.

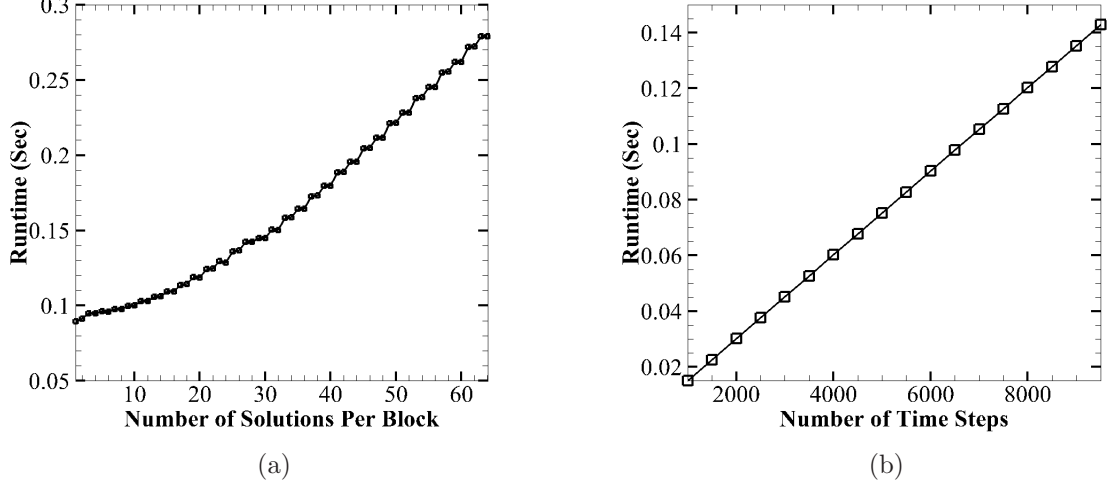


Figure 7: (a) Number of solutions per block vs. runtime in seconds; (b) Number of time points vs. runtime in seconds

effect of n_{fpb} on runtime. As the number of solutions per block increases, the runtime increases in a non-linear way. 32 solutions per block most efficiently utilize GPU resources giving forward solves with 100 millisecond runtime. Furthermore, as n_{fpb} increases beyond 32, the solutions per second gain is small while 32 solutions only requires half of the shared memory. Fig. 7(b) shows that increasing n_t causes a linear increase in runtime (as predicted by the complexity analysis in the previous section).

By appropriately choosing the number of solutions per block to ensure proper memory allocation, the runtime for a single forward solve on the GPU took 0.0151 seconds. *This is a speed-up of 32 over the CPU framework.* More importantly, the GPU based framework can execute several forwards solves simultaneously. **960 forward solves are computed in 0.0151 seconds, translating to an effective speed-up of 30000.** Effective speed-up is the direct comparison between our CPU (single-core) performance and GPU performance.

3.4. Inverse problem algorithm

The choice of the optimization algorithm is driven by the following constraints:

1. The GPU based forward solver implementation is able to compute several forward solution in parallel.
2. The parametrization of the cost function \mathcal{J} may be high dimensional.
3. Furthermore, the landscape of \mathcal{J} may be non-smooth, necessitating a gradient free method.
4. The existence of multiple local minima that have to be discarded.

A gradient-free, global search algorithm that satisfies these constraints is the particle swarm optimization (PSO) [24]. PSO finds the global minima by starting with a large population of candidate solutions (or particles), and moving these particles around in the search-space according to certain rules over the particle position and velocity. Each particle is influenced by its local best known position and the best known positions in the search-space, which are updated at every iteration as better positions are found by other particles. Generally, particle locations and velocities are chosen using a uniform distribution in the search space [25, 26]. The update of velocity uses the following equation:

$$v_i = wv_{i-1} + c_1r_1(b_l - x_{i-1}) + c_2r_2(b_g - x_{i-1}), \quad (8)$$

where v_i is the velocity at iteration i , w , c_1 , and c_2 are weighting factors, r_1 and r_2 are random numbers, b_l and b_g are the local and global bests respectively, and x_i is the position of the particle at iteration i . Recent theoretical results suggest that an appropriate choice of the parameters guarantee convergence [25, 26]. We utilize an optimized GPU based implementation of the PSO algorithm [27].

4. Results

This section discusses a hierarchy of increasingly complex inverse problems. We start with the simpler problem of extracting the base vibration characteristics given tip deflections in Section. 4.1. This problem also explores the choice of the search space parameters and their effect on runtime. The next two subsections deal with real time inversions of elastic and visco-elastic tip-sample interactions, respectively.

4.1. Base vibration force calculation

‘Experimental’ tip deflection data was computed by forcing the base to vibrate to a simple sinusoidal driven signal:

$$y(t) = a \sin(2\pi ft), \quad (9)$$

where a is the amplitude and f is the frequency of cantilever base vibration. The 'experimental' tip deflection was obtained by setting $a = 2nm$ and $f = 25600 Hz$. This 'experimental' tip deflection – subsequently used to drive the inverse problem – is obtained using the CPU based serial framework, thus resolving the issue of inverse crime [28].

We analyze the performance of the inversion framework by starting with a one dimensional search space for the amplitude of base vibrations and fixing $f = 25600 Hz$. We provide physically meaningful bounds on the amplitude, $[0 nm, 100 nm]$ and set $n_t = 1000$. The number of time steps was chosen to provide a sufficient number of data points per oscillation period given the typical time-step used. Fig. 8(a) shows that the mean and variance of the number of generation for convergence shrinks as the number of particles used in the PSO increases. Faster convergence is expected since the density of the particles in the search space is increasing. Given that the framework produces a generation about every 16 milliseconds, *the mean convergence time for the 1D case is just over 48 milliseconds for 512 particles.*

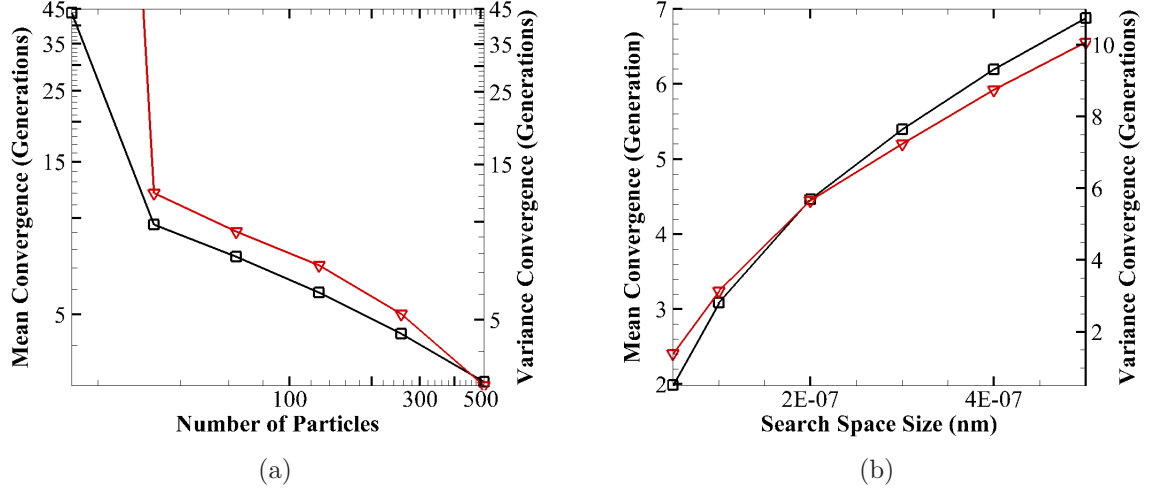


Figure 8: (a) Number of particles effect on mean(\square) and variance(∇) of generations to convergence with 1D search space; (b) Search Space effect on mean(\square) and variance(∇) of generations to convergence with 1D search space

We next investigate the affect of search space size on the performance of the inversion framework, as shown in Fig. 8(b). In general, as the particle density in the search space decreases, the mean number of generations

required for convergence increases. For the case of the largest search space ($[0 \text{ nm}, 500 \text{ nm}]$) the mean number of generation of 6.8767 corresponds to a runtime of around 112 milliseconds. *This suggests that a tight bound on the unknown force parameters can significantly decrease inversion times.*

The affect of convergence cutoff on convergence is shown in Fig. 9(a). The general trend of increasing accuracy requirements raises the mean number of generations required for convergence. To achieve an error of 0.001 requires an average of 7.99 generations (taking 128 milliseconds) verses the 48 milliseconds required for achieving a relative error of 0.01. The effect of n_t on mean generations was also tested. The number of time steps was varied from 1,000 to 10,000 and did not show any effect on the result.

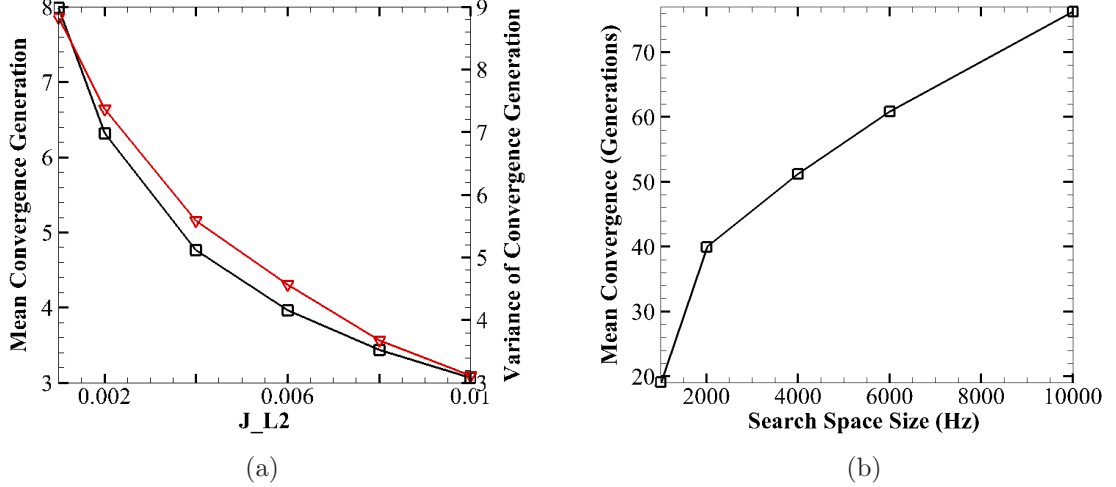


Figure 9: (a) J_{L2} target effect on mean(□) and variance(▽) of generations to convergence with 1D search space; (b) Search Space effect on mean(□) of generations to convergence with 2D search space

We next tested the inversion framework by using a 2D search space (making both a and f unknown). Fig. 9(b) shows a similar trend to the 1D case where the mean number of generations required for convergence increases as the search space increases. With a search space of 1000 Hz, 19.0458 generation on average is required for convergence resulting in a compute time of about 320 milliseconds.

To explore frequency dependent properties, the AFM can be driven by a chirp base vibration. This linearly varies the frequency of base oscillation

with time:

$$y(t) = a \sin(2\pi[f + gt]t), \quad (10)$$

where g is defined as the frequency gain parameter. We utilize the inversion framework to extract the parameters of this chirp signal. Matching a chirp base movement requires exploring a 3D search space for (a, f, g) . The 'experimental' tip deflection was obtained by setting $a = 2 \text{ nm}$, $f = 10,000 \text{ Hz}$ and $g = 20,000,000 \frac{\text{Hz}}{\text{Sec}}$. Using a search space of $[0 \text{ nm}, 100 \text{ nm}] \times [7,500 \text{ Hz}, 12,500 \text{ Hz}] \times [19,000,000 \frac{\text{Hz}}{\text{Sec}}, 21,000,000 \frac{\text{Hz}}{\text{Sec}}]$, the framework extracts correct values of (a, f, g) in 14.9 generations corresponding to a runtime of about 240 milliseconds.

4.2. Inverting elastic properties

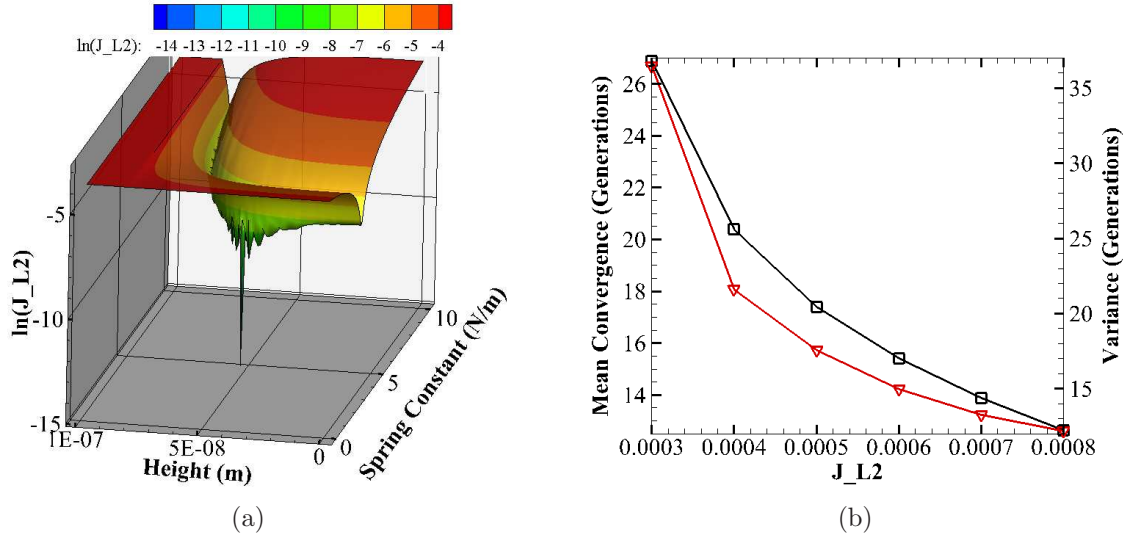


Figure 10: (a) \mathcal{J}_{L2} Space of spring and cantilever-sample separation; (b) \mathcal{J}_{L2} requirement effect on mean(\square) of generations to convergence with spring tip-sample search space

A simple yet extensively used model for tip-sample interaction is one that assumes an elastic response of the soft sample. Extracting spatial variation in elasticity is important for a variety of applications in addition to non-destructive scanning of the sample (e.g., tumors have higher stiffness than normal cells; as collagen dehydrates there is a change in elasticity [5, 14]).

We utilize a Hooke's law based parametrization for the elastic response of the sample. The modelling of the tip-sample interaction as a spring is in-line with most AFM experimental force calculation models. This model has the following form:

$$f(t) = -k(u - h) \quad (11)$$

where $u - h$ is the distance the cantilever tip has pressed into the sample and k is the tip-sample spring constant. With the base movement parameters known, only two tip-sample force parameters are unknown, the cantilever-sample separation and sample spring constant. Cantilever-sample separation is defined as distance from the cantilever's neutral axis to the sample. The 'experimental' tip deflection was created using the following parameters: $h = 50 \text{ nm}$, and $k = 4 \text{ N/m}$, where h is the cantilever-sample separation and k is the sample spring constant. These parameters were chosen to mimic results found in [15].

Calculating the force parameters for a spring sample is computationally complex due to the non-convex nature of the phase space as shown in Fig. 10(a), and results in more calculation time being required shown in Fig. 10(b). Fig. 10(a) shows that several combinations of k and h produce similar values, creating a symmetrical valley near the basin of the global minima. The resulting valley of similar combinations is expected because both k and h only affect the amplitude of vibration and not the phase. Using a cutoff convergence threshold of $\mathcal{J}_{L2} = 0.0005$ results in a 5% error in calculating k and h resulting in a mean convergence of about 18 generations, resulting in an average runtime of 288 milliseconds.

4.3. Viscoelastic tip-sample interaction

We increase the complexity of the tip-sample interaction parametrization by next assuming a visco-elastic response of the sample. Understanding the visco-elastic variations is particularly important in understanding aspects of polymer physics[29]. A Kelvin-Voigt parametrization is extensively used to model visco-elastic tip-sample interactions [15]. This is essentially a spring damper system:

$$f(x, t) = -k(u - h) - \zeta_s \dot{u} \quad (12)$$

where ζ_s is the viscous damping coefficient. We assume that the sample separation for the AFM cantilever neutral axis is known. This inversion consists of estimating two material properties. The 'experimental' tip deflections

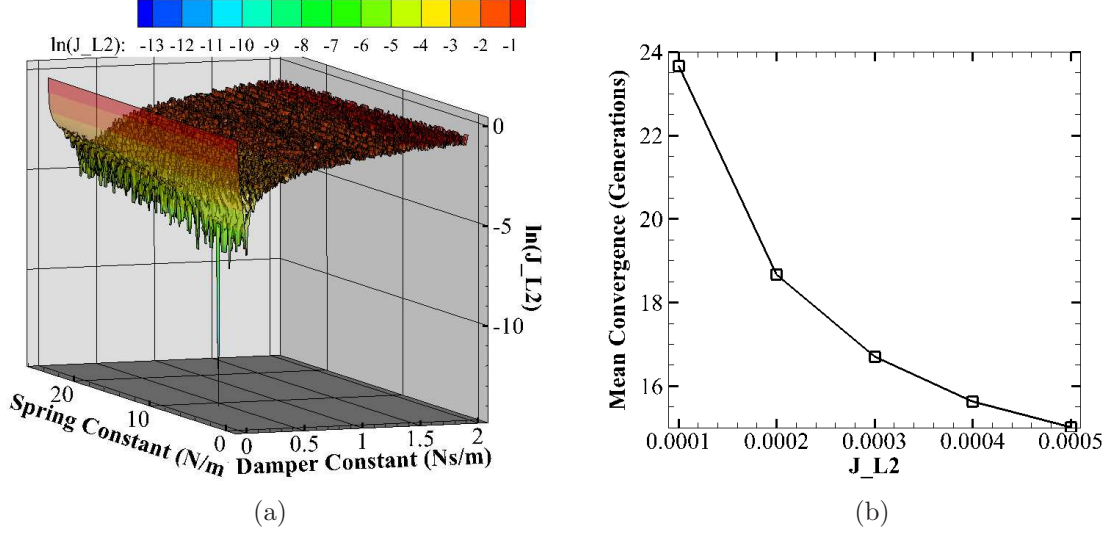


Figure 11: (a) \mathcal{J}_{L2} Space of spring and damper; (b) \mathcal{J}_{L2} requirement effect on mean(\square) of generations to convergence with spring tip-sample search space

were obtained by using the following parameters: $h = 50 \text{ nm}$, $k = 4 \text{ N/m}$, and $\zeta_s = 0.1 \frac{\text{Ns}}{\text{m}}$. Using a search space of $[0 \text{ N/m}, 25 \text{ N/m}] \times [0 \frac{\text{Ns}}{\text{m}}, 2 \frac{\text{Ns}}{\text{m}}]$ the convergence profile with increasing number of generations of the PSO scheme is shown in Fig. 11(b).

The phase space for inverting the visco-elastic tip-sample interaction is highly corrugated and has multiple local minima as shown in Fig. 11(a). The shape of search space clearly demonstrates the need for gradient free optimization methods. The large number of local minima is due to the phase change caused by the damping force. The average runtime for inversion (with 16 generations) was 256 milliseconds.

5. Conclusions

The atomic force microscope (AFM) is a versatile, high-resolution scanning tool used to characterize topography and material properties of a large variety of specimens. Its applicability to characterize soft specimens like tissue and gels is currently constrained by the inability to appropriately control tip-sample interaction forces. A major bottleneck to control tip-sample interaction is the ability to extract these tip-sample forces in real time from the deflection signal. This paper illustrates a first approach to a near real-time

framework for tip-sample force inversion. We utilize the hardware advantages and parallel capabilities of GPUs to develop a fast inversion strategy. A fast, parallel forward solver is developed that shows a 30000 fold speed-up over a comparable CPU implementation, resulting in milli-second calculation times. Posing the inverse problem as an unconstrained optimization problem allows us to integrate a GPU based gradient-free global Particle Swarm Optimization framework with the forward solver. We illustrated the framework on three classes of tip-sample interaction inversions. Each of these inversions is performed in sub-second timings showing potential for on-line integration with the AFM. Extensions of this work include (a) integrating with AFM hardware for deployment; (b) investigating more complex tip-sample parametrization that account for adhesion and Van der Waals forces; and (c) perform nano-composition mapping of soft tissue.

We envision that the developments presented here will find applicability in a broad spectrum of areas that require soft-imaging as well as nano-composition imaging. These range from nano-material synthesis and design for emerging applications such as solar cell and micro-scale energy devices; interrogating cellular and sub-cellular interactions; investigating the optoelectronic properties of organic electronic devices to understanding reaction initiation in High Energy Density Materials (HEDM). In particular, this framework should open the door to interrogate various bio-chemical interaction forces (hydrophobic interaction force) of live cell for understanding fundamental biology mechanisms such as the cell fusion process.

6. Acknowledgements

B.G. was supported in part by NSF PHY-0941576, and NSF CCF-0917202. The authors would like to thank the high-performance (GPU) computing resources at ISU.

Appendix A. Dynamic cantilever model data

L2 error	0.00333531
Length	525 microns
Elasticity	$1.76 * 10^{11}$ Pa
Width	35 microns
Height	5 microns
Inertia	$3.64583 \text{ kg } m^2$
Mass per unit length	$4.0775 * 10^{-7} \frac{kg}{m}$
Start Time	0 Sec
End Time	0.004 Sec
Time step	$4 * 10^{-8}$ Sec
Number of modes	8
Number of modal function points	501
Amplitude	5 nm
Frequency	25500 Hz
First natural frequency	25468.9 Hz
Steady State Deflection Amplitude	3.2 microns

Appendix B. Modal analysis

The deflection of the cantilever then is represented as:

$$u(x, t) = \sum_{i=1}^N \eta_i(t) \Phi_i(x), \quad (\text{B.1})$$

where $\Phi_i(x)$ represents a set of orthogonal modal shape functions which are computed by solving the homogeneous eigenvalue problem [21] and $\eta_i(t)$ are corresponding modal coefficients. In order to determine the modal coefficients, we define an inner product as:

$$\langle f, g \rangle = \int_L f g dx, \quad (\text{B.2})$$

where f and g are functions.

Appendix C. Hardware

Testing and implementation of the proposed framework occurred on a GPU mini-cluster consisting of two Dell Precision T7500 workstations. Each

node is equipped with 12 GB of DDR3 RAM, 500 GB of 10K RPM hard drives and 2 Intel Xeon 2 GHz quad-core CPUs. The main computational power comes from 4 GPUs donated to us by NVIDIA: each node is accelerated with 2 NVIDIA QUADRO FX 5800. One such card provides 240 cores, 4 GB of RAM with 102 GB/s bandwidth and is CUDA compatible (with 1.3 compute capability). The nodes are connected via dedicated Gbit Ethernet.

References

- [1] J. Loos, "The art of SPM: Scanning probe microscopy in materials science," *Advanced Materials*, vol. 17, no. 15, pp. 1821-1833, 2005.
- [2] L. G. M. Beekmans, M. A. Hempenius, and G. J. Vancso, "Morphological development of melt crystallized poly(propylene oxide) by in situ AFM: formation of banded spherulites," *European Polymer Journal*, vol. 40, no. 5, pp. 893-903, 2004. Selected papers from the 3rd International Conference on Scanning Probe Microscopy of Polymers.
- [3] J. Hahm and S. J. Sibener, "Time-resolved atomic force microscopy imaging studies of asymmetric ps-b-pmma ultrathin films: Dislocation and disclination transformations, defect mobility, and evolution of nanoscale morphology," *The Journal of Chemical Physics*, vol. 114, no. 10, pp. 4730-4740, 2001.
- [4] S. Cho, A. Quinn, M. Stromer, S. Dash, J. Cho, D. Taatjes, and B. Jena, "Structure and dynamics of the fusion pore in live cells," *Cell Biol Int.*, vol. 26, pp. 35-42., 2002.
- [5] H. Lin, D. O. Clegg, and R. Lal, "Imaging real-time proteolysis of single collagen i molecules with an atomic force microscope," *Biochemistry*, vol. 38, pp. 9956-9963, 1999.
- [6] P. K. Hansma, J. P. Cleveland, M. Radmache, D. A. Walters, P. E. Hillner, M. Bezanilla, M. Fritz, D. Vie, H. G. Hansma, C. B. Prater, J. Massie, L. Fukunaga, J. Gurley, and V. Elings, "Tapping mode atomic force microscopy in liquids," *Applied Physics Letters*, vol. 64, pp. 1738-1740, 1994.
- [7] P. Parot, Y. F. Dufre, P. Hinterdorfer, C. L. Grmellec, D. Navajas, J.- L. Pellequer, and S. Scheuring, "Past, present and future of atomic

- force microscopy in life sciences and medicine," *Journal of Molecular Recognition*, vol. 20, pp. 418-431, 2007.
- [8] J. K. H. Hörber¹ and M. J. Miles, "Scanning Probe Evolution in Biology," *Science*, vol. 302, pg. 1002-1005, 2003.
 - [9] P. P. Lehenkari, G. T. Charras, A. Nykanen, and M. A. Horton, "P. P. Lehenkari and G. T. Charras and A. Nykanen and M. A. Horton," *Ultramicroscopy*, vol. 82, pg. 289-295, 2000.
 - [10] JP Cleveland, B. Anczykowski, AE Schmid, and VB Elings, "Energy dissipation in tapping-mode atomic force microscopy," *Applied Physics Letters*, vol. 72, pg. 2613, 1998.
 - [11] R. Garcia and A. San Paulo, "Attractive and repulsive tip-sample interaction regimes in tapping-mode atomic force microscopy," *Physical Review B*, vol. 60, no. 7 pg. 4961-4967, issn. 1550-235X, 1999.
 - [12] S. Hu and A. Raman, "Chaos in atomic force microscopy," *Physical Review Letters*, vol. 96, no. 3, p. 0361072, 2006.
 - [13] C. A. V. Eysden and J. E. Sader, "Frequency response of cantilever beams immersed in viscous fluids with applications to the atomic force microscope: Arbitrary mode order," *Journal of Applied Physics*, vol. 101, pp. 044908-044918, 2007.
 - [14] F. E. Feninat, T. Ellis, E. Sacher, and I. Stangel, "A tapping mode AFM study of collapse and denaturation in dentinal collagen," *Dental Materials*, vol. 17, pp. 284-288, 2001.
 - [15] S. Hu and A. Raman, "Inverting amplitude and phase to reconstruct tipsample interaction forces in tapping mode atomic force microscopy," *Nanotechnology*, vol. 19, no. 37, p. 375704, 2008.
 - [16] W.-J. Chang, C.-M. Lin, J.-F. Lee, and S.-L. Lin, "Determination of damping force between atomic force microscope tips and sample using an inverse methodology," *Physics Letters A*, vol. 343, no. 1-3, pp. 79-84, 2005.

- [17] T. Rodriguez and R. Garcia, "Tip motion in amplitude modulation (tapping-mode) atomic-force microscopy: Comparison between continuous and point-mass models," *Applied Physics Letters*, vol. 80, num. 9, 2002.
- [18] C.A.V. Eysden and J.E. Sader, "Resonant frequencies of a rectangular cantilever beam innerised in a fluid," *Journal of Applied Physics*, vol. 100, Dec 2006.
- [19] J. Sader, "Frequency response of cantilever beams immersed in viscous fluids with applications to the atomic force microscope," *Journal of Applied Physics*, vol. 84, no. 1, pg. 64-76, 1998.
- [20] J. W. M. Chon, P. Mulvaney, and J. Sader, "Experimental validation of theoretical models for the frequency response of atomic force microscope cantilever beams immersed in fluids," *Journal of Applied Physics*, vol. 87, no. 8, pg. 3978-3988, 2000.
- [21] L. Meirovitch, *Fundamentals of Vibrations*. 1221 Avenue of the Americas, New York, NY, 10020: McGraw-Hill, 2001.
- [22] V. Morris, A. Kirby, and A. Gunning, *Atomic Force Microscopy for Biologists*, 57 Shelton Street, Covent Garden, London, WC2H 9HE, United Kingdom: Imperial College Press, 2010.
- [23] "CUDA toolkit reference manual," Website, 2010.
http://developer.download.nvidia.com/compute/cuda/3_2/toolkit/docs/CUDA_Toolkit_3.2.pdf
- [24] J. Kennedy and R. Eberhart, "Particle swarm optimization," *IEEE International Conference on Neural Networks Proceedings*, vol. 4, pg. 1942 - 1948, 1995.
- [25] I. C. Trelea, "The particle swarm optimization algorithm: convergence analysis and parameter selection," *Information Processing Letters*, 85, pp. 317-325, 2003.
- [26] M. Jiang, Y.P. Luo, and S.Y. Yang, "Stochastic convergence analysis and parameter selection of the standard particle swarm optimization algorithm," *Information Processing Letters*, 102, pp. 8-16, 2007.

- [27] L. Mussi, F. Daolio, and S. Cagnoni, "CUDA-PSO," Version 1.0, Universita' degli Studi di Parma, Italy, 2011, http://www.ce.unipr.it/people/mussi/projects/CUDA-PSO-v1.0-html_documentation/.
- [28] J. Kaipio and E. Somersalo, *Statistical and Computational Inverse Problems*, 233 Spring Street, New York, NY, 10013, USA: Springer Science and Business Media, Inc., 2005.
- [29] R. Garcia, C.J.Gomez, N.F.Martinez, S. Patil, C. Dietz, and R. Magerie, "Identification of Nanoscale Dissipation Processes by Dynamic Atomic Force Microscopy," *Physics Review Letters*, vol. 97, issue 1, 2006.